Descoperirea cunoştinţelor conceptuale în baze de date

Rezumat al tezei de doctorat

Katalin Tünde Jánosi-Rancz

Şcoala Doctorală de Matematică şi Informatică

Conducător:

Prof. Dr. Zoltán Kása

Departmentul de Informatică

Facultatea de Mathematică şi Informatică

Universitatea Babeş-Bolyai

Cluj-Napoca, România

2014

Lista publicaţiilor

1. **K.T. Janosi-Rancz**, A. Lajos, Semantic Data Extraction, Accepted for The 8th International Conference INTER-ENG 2014, Interdisciplinarity in Engineering, 9 - 10 October 2014, "Petru Maior" University of Tîrgu Mureş, Romania

2. **K.T. Janosi-Rancz**, Finding, Managing and Inforcing CFDs and Ars via a semi-automatic learning strategy, 10th Joint Conference on Mathematics and Computer Science, May 21-25, Cluj Napoca, Romania, 2014

3. **K.T. Janosi-Rancz**, V. Varga, XML Schema Refinement Through Formal Concept Analysis, Studia Universitatis Babes-Bolyai, Informatica, Volume LVII, Number 3, 2012, 49-64

4. V. Varga, **K. T. Janosi Rancz**, C. Sacarea, K. Csioban, XML Design: an FCA Point of View, Proceedings of 2010 IEEE International Conference on Automation, Quality and Testing, Robotics (AQTR), Theta 17th edition, 2010, 165-170

5. **K. T. Janosi-Rancz**, V. Varga and T. Nagy, Detecting XML Functional Dependencies through Formal Concept Analysis, 14th East European Conference on Advances in Databases and Information Systems (ADBIS), Novi Sad, Serbia, Springer LNCS 6295, 2010, 595-598 **(Citat de 2 ori)**

6. D. Ignatov, **K. T. Janosi-Rancz**, S. Kuznetsov, Towards a framework for near-duplicate detection in document collections based on closed sets of attributes. Acta Universitatis Sapientiae, Informatica, 2/2009, 215-233

7. **K. T. Janosi Rancz**, V. Varga, J. Puskas, A Soft Tool for Relational Database Design using Formal Concept Analysis, 7th Joint Conference on Mathematics and Computer Science, Studia Universitatis Babes-Bolyai, Informatica, vol. LIII, No. 2, 2008, 67-78 **(Citat de 1)**

8. **K. T. Janosi Rancz**, V. Varga, A method for mining functional dependencies in

relational database design using FCA. Studia Universitatis Babes-Bolyai, Informatica, vol. LIII, No. 1, 2008, 17-28 **(Citat de 6 ori)**

9. V.Varga, **K. T. Janosi Rancz**, A Software Tool to Transform Relational Databases in Order to Mine Functional Dependencies in it Using Formal Concept Analysis, Proceedings of the Sixth Int. Conf. on Concept Lattices and Their Applications (CLA), Olomouc, Czech Republic, 2008, 1-9 **(Citat de 2 ori)**

Cuvinte cheie: Knowledge Discovery, XML design, Formal Concept Analysis, Semantic Search, Semantic Set, Data Mining, Information Extraction, Dependency Mining, Association Rule, Dependency Management, Concept Lattice, Data Management, Optimization, Data Extraction, Information retrieval

Cuprinsul tezei

Conţinutul rezumatului

1 Introducere

1.1 Descoperirea cunoștințelor conceptuale în baze de date

Odată cu creşterea de fără precedent a generării şi acumulării datelor on-line, a crescut şi dependenţa noastră de baze de date în privinţa stocării acestor date. Există o necesitate urgentă pentru noi metode de abordare a acestei provocări de lungă durată. Cum putem identifica, extrage şi prezenta în mod eficient informaţiile utile care sunt incluse în aceste baze de date? Şi mai ales cum putem să le prezentăm într-un mod benefic nevoilor actuale şi viitoare ale utilizatorului final? Găsirea răspunsurilor la aceste întrebări a fost sursa noastră de inspiraţie pentru a dezvolta abordările care sunt prezentate în această teză.

Descoperirea cunoştinţelor în baze de date (Knowledge Discovery in Databases - KDD) [35] are ca scop dezvoltarea unor metode, tehnici şi instrumente care ajută analiştii umani în procesul general de a obţine unităţi de cunoaştere care la rândul lor pot fi utilizate în rezolvarea problemelor reale. Conceptele sunt necesare pentru a exprima cunoaşterea umană [81]. În mod recent teoria laticelor a dus în gândirea matematică la reprezentarea cunoaşterii şi descoperirea cunoştinţelor. Analiza formală a conceptelor (Formal Concept Analysis - FCA) [80; 26; 7; 10; 58; 44] oferă o abordare naturală pentru a descoperi concepte formale în datele care sunt descrise în forma unui context formal, descoperirea dependenţelor de date şi vizualizarea acestora printr-o singură structură conceptuală numită laticea conceptuală. Reprezentarea grafică a laticelor conceptuale s-a dovedit a fi extrem de utilă în descoperirea şi înţelegerea relaţiilor conceptuale într-un anumit set de date. Laticea conceptuală s-a dovedit a fi utilă şi în alte aplicaţii referitoare la prelucrarea informaţiilor şi a cunoştinţelor cum ar fi vizualizarea, analiza datelor, mineritul datelor (data mining) şi managementul cunoştinţelor.

FCA asigură suport în aşa numita descoperire a cunoştinţelor conceptuale în baze de date (Conceptual Knowledge Discovery in Databases - CKDD) care are ca scop facilitarea unui proces centrat pe om, de a descoperi cunoştinţe în date prin vizualizarea şi analiza structurii conceptuale a datelor [39].

Scopul nostru a fost de a extrage cunoştinţe conceptuale de înalt nivel din baze de date. Contribuţia noastră principală este mineritul diferitelor tipuri de dependenţe de date (în date relaţionale şi XML) utilizând FCA; ne concentrăm, de asemenea, pe managementul dependenţelor, prevenirea incoerenţelor şi extragerea de date semantice. Toate cercetările noastre sunt efectuate în domeniul CKDD.

1.2 Contribuţii şi structura tezei

În acestă secţiune vom prezenta principalele noastre contribuţii şi modul în care acestea sunt incluse în structura tezei. Contribuţia noastră principală se referă la mineritul diferitelor tipuri de dependenţe de date (în date relaţionale şi XML) utilizând analiza formală a conceptelor; ne concentrăm, de asemenea, pe managementul dependenţelor, prevenirea incoerenţelor şi extragerea de date prin prezentarea modului în care abordările noastre contribuie la descoperirea de cunoştinţe conceptuale. În acest rezumat numerotarea capitolelor urmează cea a tezei.

În capitolul 1 introducem principalele contribuţii şi modul în care acestea sunt structurate în teză.

În capitolul 2 prezentăm fundamentele teoretice, în capitolul 3 prezentăm modul în care FCA poate fi utilizată pentru a caracteriza dependenţe funcţionale (FD-uri). O primă problemă în utilizarea de FCA pentru a obţine dependenţele funcţionale este faptul că seturile de date sunt în general multivalorice şi nu binare. Acest lucru înseamnă că setul de date trebuie transformat într-un fel pentru obţinerea unui context binar ale cărui implicaţii sunt echivalente cu aceste dependenţe. Din păcate acest lucru duce la o nouă reprezentare a datelor, una pătratică în numărul obiectelor referitoare la datele originale, care nu permit aplicarea acestei metode pe seturi de date de mari dimensiuni. Acesta a fost motivaţia noastră de a crea o metodă de construire a unor fişiere index sau inversate pentru a optimiza construirea contextului formal al dependenţelor funcţionale. Metoda noastră permite o caracterizare echivalentă dar cu proprietăţi de calcul mai bune. În acest capitol prezentăm software-ul nostru (FCAFuncDepMine) care poate fi utilizat în proiectarea bazelor de date relaţionale şi în detectarea dependenţelor funcţionare în tabele existente. Acest software poate fi considerat un mediu de asistenţă a descoperirii cunoştinţelor conceptuale, unul care promovează o abordare centrată pe om a proceselor de descoperire a dependenţelor şi de reprezentare a cunoştinţelor obţinute.

În capitolul 4 propunem un cadru în care documentul XML este analizat, iar contextul formal care corespunde reprezentării plane a datelor XML este construit. Apoi este analizat

din punctul de vedere al descoperirii cunoştinţelor. Laticea conceptuală astfel obţinută este o reprezentare grafică utilă a elementelor incluse în documentul XML analizat şi a ierarhiei lor. Tot în acest capitol este introdus software-ul nostru (FCAMineXFD). Acest software găseşte cheile şi dependenţele funcţionale în datele XML, fiind implicaţii atribuite în contextul formal construit. Schema documentului XML este transformat în GTT-XNF prin utilizarea dependenţelor funcţionale identificate.

În capitolul 5 introducem abordarea noastră referitoare la managementul dependenţelor şi prevenirea contradicţiilor. Descriem strategia noastră de a găsi dependenţe funcţionale condiţionate (Conditional Functional Dependencies - CFD-uri) [30] şi reguli de asociere (Association Rules – AR-uri) [2], şi în loc de a le utiliza pentru a curăţa "date murdare" (dirty data) le folosim pentru a preveni apariţia lor în baza de date. Acest lucru a fost realizat prin diferenţierea dependenţelor stricte de cele aparente. Dacă o dependenţă este strictă, se iau în considerare toate posibilele excepţii de la dependenţa invalidă, prin urmare la momentul validării sunt generate condiţii care previn inserţii şi actualizări (update) care nu sunt conforme cu dependenţa validată. Numărul modelelor poate fi foarte mare, de aceea am introdus o relaţie de ordine parţială "mai utilă" (More Useful - MU), şi în fiecare caz de validare sunt luate în considerare supremurile, adică dependenţele cele mai utile. Relaţia de ordine parţială a MU este extrem de utilă deoarece previne inundarea bazei de date cu dependenţe valide care sunt consecinţele ale unor dependenţe mai utile. Abordarea de a preveni apariţa de date murdare în baza de date ajută în menţinerea consistenţei datelor stocate. Împreună cu managementul complet al dependenţelor, aplicaţia noastră implementată, DependencyManager, utilizează metode de Analiza Formală a Conceptelor pentru a analiza dependenţele stricte, şi pentru a trage concluzii importante, astfel ajutând utilizatorii în prevenirea neconcordanţelor, în repararea erorilor şi optimizarea interogărilor şi aplicaţiilor prin furnizarea unei latice de dependenţe, utilizând utilitatea ca şi relaţie.

În capitolul 6 prezentăm instrumentul nostru care ajută extragerea datelor din diferite tipuri de surse de date. Extractorii de date sunt în general greu de întreţinut pentru că trebuie să facă faţă unor structuri de date posibil infinite şi a unor moduri posibil infinite pentru a descrie acelaşi lucru. Pentru a simplifica sarcina dificilă de a integra noi surse de date şi de a lucra cu schimbări structurale în sursele de date existente am implementat o nouă abordare denumită *RK*. În *RK* arborele semantic generat de regulile semantice (utilizate ca şi date de intrare) descrie un model care reprezintă structura sursei de date. Acest instrument este fezabil aşa cum este, dar potenţialul maxim este atins când se creează o colaborare între proprietarul sursei de date şi extractorul, regulile semantice putând fi actualizate ori de câte ori structura sursei de date este schimbată. Acest lucru ar face instrumentul imun la schimbările structurale. Facilitarea extragerii semantică a datelor ar facilita şi căutările semantice pe baza datelor extrase. Rezultatele experimentului în care s-a folosit instrumentul RK au dovedit fezabilitatea abordării noastre prin faptul că extragerea a fost amplificată din punctul de vedere al preciziei şi al vitezei.

1.3 Contribuţii publicate anterior

Rezultatele prezentate în această teză au fost deja publicate. Metoda noastră de construire a unor fişiere index sau inversate pentru a optimiza construirea contextului formal al dependenţelor funcţionale a apărut în [73; 44]. Software-ul nostru, FCAFuncDepMine de extragere a dependenţelor funcţionale care deţin un set de date prin folosirea de FCA poate fi găsit în [48].

Cadrul nostru care oferă o vizualizare interactivă a explorării dependenţelor în date XML poate fi găsit în [47; 43; 74]. Managementul dependenţelor şi prevenirea incoerenţelor sunt, de asemenea, prezentate în [46]. Instrumentul nostru care ajută extragerea datelor din diferite tipuri de surse de date a fost publicat în [45].

2

Fundamentele teoretice

În capitolul al doilea al tezei prezentăm fundamentele teoretice: conceptele de baza ale FCA: ordine şi latice, context, concept şi laticea aferentă, cu exemple şi figuri aferente. Scalarea conceptuală şi dependenţele în baze de date relaţionale: funcţionale, condiţional funcţionale şi reguli de asociere, dependenţele în XML, sunt redate prin definiţiile corespunzătoare, exemple şi figuri.

3

Caracterizarea dependenţelor funcţionale (FD) cu contexte formale

În acest capitol prezentăm prima noastră contribuţie care se referă la dezvoltarea abordărilor optimizate de extragere FD cu utilizare FCA. Rezultatele prezentate în acest capitol au fost publicate în [73; 44; 48].

O primă problemă în utilizare FCA pentru a calcula FD este faptul că seturile de date sunt în general multivalorice şi nu binare. Acest lucru înseamnă că setul de date trebuie transformat într-un fel pentru obţinerea unui context binar ale cărui implicaţii sunt echivalente cu aceste dependenţe. Hereth în [38] oferă o soluţie pentru această binarizare. Din păcate în cazul metodei descrise în [18] numărul obiectelor contextului rezultat este pătratic referitor la original, astfel această metodă nu poate fi aplicată pe seturi de date de mari dimensiuni. Aceasta a fost motivaţia noastră de a crea o metodă de construire a unor fişiere index sau inversate pentru a optimiza construirea contextului formal al dependenţelor funcţionale. Ne-am propus să descoperim cunoştinţe conceptuale, totuşi scopul principal a fost de a reduce la minim datele de intrare înainte de a aplica FCA. Am optimizat metoda lui Hereth prin reducerea semnificativă a fişierului de context.

În continuare prezentăm metoda noastră de a construi contextul dependenţelor funcţionale al unei baze de date tabel.

În procedura de creare a contextului excludem perechile de tuple care sunt simetrice şi reflexive pentru a evita redundanţele. În partea superioară a laticei conceptuale se vor afla perechi de tuple în care nu sunt valori comune pentru atributele corespunzătoare, astfel putem omite generarea acestor perechi de tuple. Perechile cu forma (t; t), unde t este un

tuplu al tabelului, au toate atributele în comun. Aceste obiecte vor ajunge în partea de jos a laticei, astfel putând fi omise, fiindcă nu schimbă aceste implicaţii în context. Pentru a optimiza construirea contextului formal al dependenţelor funcţionale am construit fişiere index sau inversate pentru valorile fiecărui atribut. Cu aceste indice inversate numărul rândurilor în fişierul de date al contextului formal care rezultă în urma aplicării metodei noastre este jumătatea valorii care rezultă în urma aplicării metodei lui Hereth în [38]. În consecinţă putem reduce timpul necesar pentru a construi laticea conceptuală pentru dependenţe funcţionale şi putem elimina dependenţele inutile.

3.1 Rezultatele experimentului

Pentru a evalua metoda noastră de creare a contextului prin folosirea indicelor inversate am efectuat experimente pe diferite seturi de date.
Se consideră următoarea tabelă incorect concepută:

```
StudMarks [StudID, StudName, GroupID, Email, SpecID,
 SpecName, Language, DiscID, DName, CreditNr, Mark]
```

Această tabelă conţine 19500 de rânduri şi 11 coloane; metoda noastră a redus numărul obiectelor de la 380.250.000 la 110.073.342 în fişierul de context.

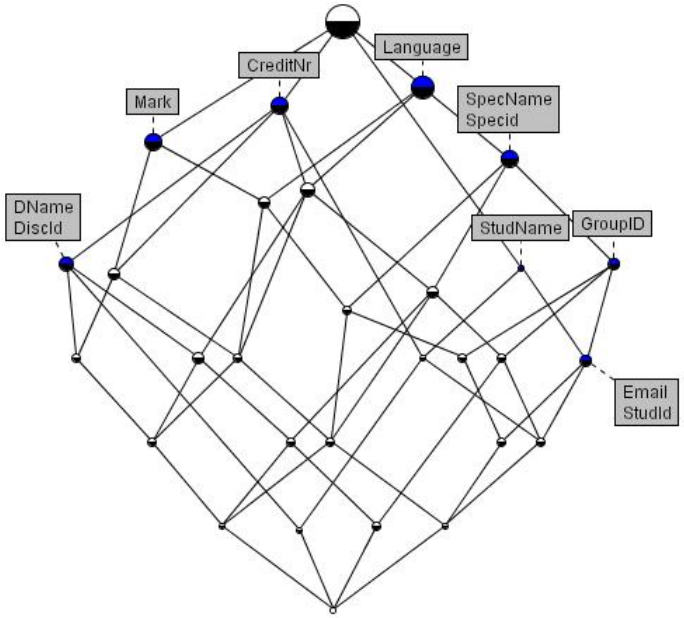

Fig. 3.1: Laticea conceptuală $FD\left(StudMarks, \overrightarrow{\mathbb{K}}(Uni)\right)$

Am aplicat FCA pe contextul redus; Figura 3.1 include laticea conceptuală rezultată în cazul tabelei StudMarks. Implicaţiile în această latice, care sunt dependenţe funcţionale în tabelă, pot fi interpretate în următorul fel: conceptul cu denumirea $GroupID$ este un

subconcept al conceptului cu denumirea $SpecID$ şi $SpecName$. Acest lucru înseamnă că în fiecare pereche de tuple, unde câmpul $GroupID$ are aceeaşi valoare, specializarea ID şi numele este identic. Astfel avem următoarele implicaţii, care sunt dependenţe funcţionale:

$$GroupID \rightarrow SpecID,\ SpecName$$

În acelaşi mod următoarele dependenţe funcţionale pot fi citite din laticea conceptuală:

$$DiscID \rightarrow CreditNr$$
$$DName \rightarrow CreditNr$$
$$SpecID \rightarrow Language$$
$$SpecName \rightarrow Language$$
$$StudID \rightarrow StudName$$
$$Email \rightarrow StudName$$

De asemenea am efectuat experimente utilizând bine-cunoscutele seturi de date din baza de învăţare automată UCI http://archive.ics.uci.edu/ml/, şi anume seturile de date Wisconsin Breast Cancer (WBC) şi Mushroom. Putem concluziona că prin utilizarea acestor indice inversate, numărul rândurilor în fişierul de date al contextului formal care rezultă în urma aplicării metodei noastre este jumătatea valorii care rezultă în urma aplicării metodei lui Hereth în [38]. Tabelul 3.1 include sumarul performanţei metodei noastre. Astfel putem reduce timpul necesar pentru a construi laticea conceptuală pentru dependenţe funcţionale şi putem elimina dependenţele inutile. La acest punct subliniem faptul că prin optimizarea dimensiunii contextului calitatea setului de dependenţe creat nu s-a deteriorat; metoda noastră permite o caracterizare echivalentă dar cu proprietăţi de calcul mai bune. Am implementat metoda prezentată în acest capitol şi l-am completat cu un instrument de software care analizează tabelele de date relaţionale existente.

Dataset	Arity	No. Rows	Nr. obiectelor cu metoda noastra	Nr. obiectelor cu metoda lui Hereth [38]
Mushroom	22	8,124	32,995,626	65,999,376
WBC	11	699	210,934	481,636
StudentMarks	11	19,500	110,073,342	380,250,000

Table 3.1: Rezultate experimentale: contextul optimizat

Software-ul nostru denumit FCAFuncDepMine, construieşte un context formal optimizat de dependenţe funcţionale. Software-ul poate fi utilizat în proiectarea bazelor de date relaţionale şi în detectarea dependenţelor funcţionare în tabele existente. În prezent FCAFuncDepMine suportă trei tipuri de format de fişiere de date de intrare: XML, format csv şi tabele de baze de date relaţionale. FCAFuncDepMine salvează contextul formal în format .cex utilizat pe scară largă.

Optimizările au loc în FCAFuncDepMine în mod adecvat pentru a transforma seturi de date de mari dimensiuni în contexte formale. Software-ul nostru poate fi considerat un mediu de asistenţă a descoperirii de cunoştinţe conceptuale, care promovează o abordare centrată pe om a proceselor de descoperire, a dependenţelor şi de reprezentare a rezultatelor.

4

Dependenţe funcţionale XML cu FCA

Ca o a doua contribuţie propunem un cadru în care este analizat documentul XML, şi este construit contextul formal, care corespunde reprezentării plane a datelor XML. Rezultatele prezentate în acest capitol au fost publicate în [47; 43; 74]. Metoda noastră este suportată de un cadru denumit FCAMineXFD care găseşte cheile şi dependenţele funcţionale în datele XML, fiind implicaţii atribuite în contextul formal construit. Laticea conceptuală obţinută este o reprezentare grafică utilă a elementelor incluse în documentul XML analizat şi a ierarhiei lor. Schema documentului XML este transformată în GTT-XNF [90] prin utilizarea dependenţelor funcţionale identificate.

Pentru a realiza acest lucru datele XML trebuie convertite în contexte formale FCA. Software-ul nostru poate analiza întregul document XML sau o clasă tuplu C_p dat de traiectoria p. Datele XML pot fi convertite într-o relaţie total neimersată, o singură tabelă relaţională şi algoritmii de descoperire FD existenţi pot fi aplicaţi în mod direct. Dintr-un document XML care conţine schema datelor de la început, creăm arbori de tupluri generalizate.

Ca un prim pas trebuie să definim obiectele şi atributele de interes şi să creăm modele de XML în termeni de context.

- Alegerea atributelor FCA: *PathEnd/ElementName*

  – în cazul nodurilor interne (non-leaf) numele atributului este construit în următorul fel:
  <ElementName>+"@key" " şi valoarea lui va fi valoare cheii asociate. Mai multe

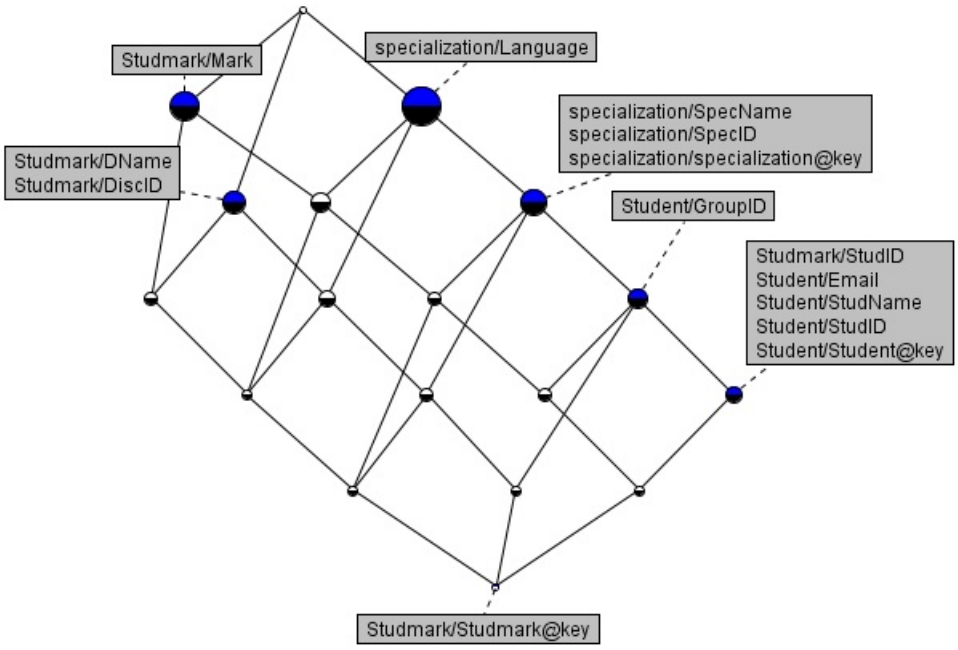

Fig. 4.1: Laticea conceptuală pentru $C_{specialization}$

elemente care au aceeaşi traiectorie, vor avea acelaşi nume de atribut, dar valorile vor fi diferite.

– frunzele (nu valorile pe frunze, ci numele elementelor pe frunze) arborelui tuplu.

- Alegerea obiectelor: obiectele sunt considerate perechile tuplu ale arborelui, mai precis perechile de tuplu din tabela plată. Valorile cheie asociate elementelor interne (non-leaf) şi valorile elementelor leaf sunt utilizate în aceste perechi de tupluri.

Documentul XML analizat poate avea un număr mare de arbori tuplu. Folosim aceleaşi tehnici de optimizare ca şi în capitolul anterior, filtrăm perechile de tupluri şi omitem acele perechi care nu au atribute comune, ceea ce nu modifică ierarhia conceptuală.

Odată ce obiectele şi atributele contextului sunt definite, generăm conceptele şi creăm laticea conceptuală în acelaşi mod ca şi în cazul bazelor de date relaţionale. O latice conceptuală pentru o bază de date XML universitară (clasa tuplu $C_{specialization}$) se poate observa în figura 4.1, unde putem interpreta fiecare concept şi genera lista tuturor dependenţelor funcţionale.

Example 4.1. *În nodul conceptual cu eticheta specialization/specialization@key, specialization/SpecID, specialization/SpecName obiectele asociate sunt perechile de tupluri, unde valorile pentru specialization=SpecID sunt aceleaşi. Deci avem următoarele FD-uri XML:*

$$\langle C_{specialization}, ./SpecID, ./SpecName \rangle$$

$$\langle C_{specialization}, ./SpecID, ./specialization@key \rangle$$

$$\langle C_{specialization}, ./specialization@key, ./SpecID \rangle$$

$$\langle C_{specialization}, ./specialization@key, ./SpecName \rangle$$

$$\langle C_{specialization}, ./SpecName, ./specialization@key \rangle$$

$$\langle C_{specialization}, ./SpecName, ./SpecID \rangle$$

Din laticea conceptuală se poate citi relaţia dintre concepte. Există 1:N relaţii de la *Specialization* la *Group*, de la *Group* la *Students*, de la *Students* la *Studmark*. *Disciplines* este într-o relaţie N:N cu *Students*, conectat la nodul *Studmark* în acest caz. În abordarea noastră pe lângă XFD-uri putem găsi cheile XML pentru un document XML dat.

Implicaţiile găsite de FCAMineXFD conţin unele FD-uri cu RHS ca şi valori cheie (./@key). Acestea pot fi utilizate pentru a detecta cheile în XML. Acesta este un pas important în descoperirea de cunoştinţe bazate pe FCA, deoarece până acum au putut fi citite doar dependenţele din laticea conceptuală, dar prin utilizarea metodei noastre putem găsi şi cheile.

Sunt de exemplu două FD-uri (Ex. 4.1) cu RHS ca şi $./specialization@key$ în clasa tuplu $C_{specialization}$, astfel cheile XML detectate sunt:
$\langle C_{specialization}, ./SpecID \rangle$, $\langle C_{specialization}, ./SpecName \rangle$.

Dat fiind setul de chei şi setul de dependenţe descoperite de instrumentul nostru, vom converti schema XML într-una corectă.

Ca o a doua contribuţie propunem un cadru în care este analizat documentul XML, este construit contextul formal care corespunde reprezentării plane a datelor XML şi este efectuată analiza acestuia din punctul de vedere al descoperirii de cunoştinţe. Aşa cum am observat laticea conceptuală obţinută în aşa fel este o reprezentare grafică valoroasă a elementelor incluse în documentul XML analizat, şi a ierarhiei lor. Software-ul nostru, FCAMineXFD, găseşte cheile şi dependenţele funcţionale între datele XML, fiind implicaţii atribuite în contextul formal construit. Schema documentului XML este transformat în GTT-XNF prin utilizarea dependenţelor funcţionale identificate.

5

Managementul dependenţelor şi prevenirea incoerenţelor

În acest capitol descriem a treia contribuţie care găseşte dependenţe funcţionale condiţionate (Conditional Functional Dependencies - CFDs) [30] şi reguli de asociere (Association Rules - ARs) [2] şi în loc de a le utiliza pentru a curăţa "date murdare" (dirty data) le folosim pentru a preveni apariţia acestora în baza de date. Acest lucru a fost realizat prin diferenţierea dependenţelor stricte de cele aparente. Dacă ştim despre o dependenţă că va deveni validă în viitor, ne putem baza pe acestea prin crearea unor limite care garantează că regula CFD nu va fi încălcată de inserţii sau modificări. Împreună cu managementul complet al dependenţelor aplicaţia noastră implementată denumită DependencyManager, utilizează metode de Analiza Formală a Conceptelor pentru a analiza dependenţele stricte şi trage concluzii importante, astfel ajutând utilizatorii în prevenirea neconcordanţelor, în repararea erorilor şi optimizarea interogărilor şi aplicaţiilor prin furnizarea unei latice de dependenţe, folosind utilitatea ca şi relaţie. Managementul dependenţelor şi prevenirea incoerenţelor sunt, de asemenea, prezentate în [46].

Aplicaţia noastră învaţă FD-uri, CFD-uri şi AR-uri dintr-o bază de date arbitrară. Putem valida dependenţele învăţate prin acceptarea sau respingerea lor, astfel modificând datele de la candidat la dependenţe stricte (SD) sau aparente (AD). Am creat sprijin pentru simplificarea validării prin utilizarea relaţiei MU astfel încât cei mai utili candidaţi să fie luaţi în considerare dintre toate celelalte dependenţe, de aceea reducem în mod drastic numărul dependenţelor care trebuie luate în considerare şi păstrăm doar cele mai utile SD-uri deoarece toate celelalte dependenţe sunt implicate de acestea. Rezultatul cercetării noastre este un

sistem care este capabil de a preveni toate incoerenţele posibile care rezultă din inserţii sau modificări (update) care nu sunt conforme schemei SD-urilor acceptate prin utilizarea limitelor. Folosim metode FCA pentru a analiza CFD-urile/AR-urile stricte şi pentru a trage concluzii utile. Cu alte cuvinte: exploatăm FCA-ul pentru a obţine concepte care conţin ontologie.

Strategia noastră se bazează pe învăţarea automată, validarea semi-automată şi uitarea automată. Am definit un algoritm care este folosit de aplicaţie pentru a învăţa noi CFD-uri şi AR-uri. Acceptarea automată a tuturor dependenţelor învăţate poate fi riscantă mai ales dacă datele sunt critice. Validarea nu este întotdeauna uşoară pentru utilizatori, deoarece pot fi multe dependenţe învăţate, dar am utilizat relaţia "mai util" (MU), care reduce semnificativ dificultatea validării dacă este folosită în mod corespunzător. Aşa cum am mai menţionat, utilizatorul vede o listă cu cele mai utile CFD-uri/AR-uri şi poate decide dacă acestea sunt SD-uri sau AD-uri. Dacă utilizatorul acceptă o dependenţă D_1 atunci toate D_2 vor deveni instantaneu invalizi ceea ce îndeplineşte criteriile de D_1 MU D_2. Ca urmare, orice D_2 mai puţin util decât D_1 este redundant cu D_1. Dacă utilizatorul respinge un CFD/AR D_1, atunci acesta va deveni invalid iar toate CFD-urile/AR-rile D_2 care îndeplinesc criteriile

$$(D_1 \text{ MU } D_2) \wedge (\nexists D_3 \text{ astfel ca } D_1 \text{ MU } D_3 \text{ MU } D_2)$$

vor fi afişate utilizatorului deoarece vor deveni cele mai utile AR-uri/CFD-uri.

Am estimat beneficiile de utilizare a relaţiei MU. Dacă o tabelă are n coloane şi o dependenţă are o coloană pentru condiţia, s coloane în setul de coloane determinante, şi d coloane în setul de coloane dependente, există $2^{(n-s-d-1)} + 2^d - 1$ posibile dependenţe care sunt mai puţin utile. Dacă acceptăm dependenţa dată, în mod automat refuzăm toate dependenţele mai puţin utile, ceea ce optimizează, ajutând sistemul/utilizatorul cu reducerea automată a dependenţelor care trebuie luate în considerare cu un maxim de $2^{(n-s-d-1)}+2^d-1$ dependenţe. Această metodă ajută mult în validarea şi în utilizarea dependenţelor. Aceasta este o optimizare exponenţială.

5.1 Rezultatele experimentului

Am evaluat eficienţa şi eficacitatea algoritmului nostru pe patru seturi de date. În seturile de date denumite Numbers1 şi Numbers2 rândurile au fost numere generate folosind formule pentru a garanta apariţia CFD-urilor şi AR-urilor. De asemenea am efectuat experimente utilizând seturi de date reale din baza de învăţare automată UCI http://archive.ics.uci.edu/ml/, şi anume seturile de date Wisconsin Breast Cancer (WBC). Setul de date Exceptions a fost, de asemenea, generat de noi, prin folosirea datelor reale.

Tabelul 5.1 descrie parametrii seturilor de date utilizate, rata MinOccurrenceFrequency şi MinOccurrenceRate utilizată în experimentul nostru şi numărul de dependenţe accep-

Dataset	Arity	No. Rows	MOF	MOR	SD	AD
Numbers1	6	20000	200	0.15	95	13
Numbers2	5	200000	650	0.2	118	11
WBC	11	699	37	0.06	242	21
Exceptions	5	11192	1000	0.1	43	8

Table 5.1: Rezultatele experimentului (MOF = MinOccurrenceFrequency, MOR = MinOccurrenceRate)

tate/respinse: SD-uri şi AD-uri pentru fiecare set de date.

Pe baza analizei noastre am reuşit să reducem semnificativ numărul dependenţelor luate în considerare, am constatat că doar 10% -30% dintre dependenţe au fost utile.

În cazul în care un model SD este acceptat (validat), o limitare împiedică inserţiile şi actualizările incoerente cu modelul acceptat. Am cuantificat cantitatea de date protejate de sistemul nostru împotriva încălcării regulii de incoerenţă. În setul nostru de date Numbers1 8620, Numbers2 102034, BCW 384, Exceptions 7083 reprezintă numărul de înregistrări care îndeplinesc cel puţin un SD, cu o inserţie sau o actualizare arbitrară în viitor având o probabilitate de 43,1%, 51,01%, 54,93% şi 63,28% de a fi protejat prin sistemul nostru împotriva incoerenţelor care încalcă SD-urile noastre. Dacă apare o eroare din cauza manipulării necorespunzătoare a datelor unde un SD este aplicabil, problema poate fi identificată în jurnalele de eroare şi, în mod potenţial, erorile pot fi găsite şi reparate.

În DependencyManager am implementat un modul FCA, care studiază proprietăţile SD-urilor descoperite cu ajutorul aplicaţiei noastre şi trage concluzii interesante şi cunoştinţe adiţionale despre ele. Abordarea noastră a fost agnostică faţă de natura conţinutului bazei de date; am implementat un FCA aplicabil dependenţelor acceptate ale fiecărei baze de date prin utilizarea sistemului nostru. Ca un prim pas am transformat cunoştinţele existente într-un context FCA bazat pe SD-uri. În acest context obiectele sunt FD-uri, iar atributele sunt proprietăţile posibile ale obiectelor. Atributele sunt Weakness (Slăbiciune), Determinant Column Cluster's Size (dimensiunea fasciculului coloanei determinante), Column Cluster Size (dimensiunea fasciculului coloanei), Frequency of Occurence (frecvenţa apariţiei) şi Almost Symmetrical (aproape simetric). Am determinat seturile fuzzy de None (niciunul), Very Low (foarte scăzut), Low (scăzut), Medium (mediu), High (mare), Very High (foarte mare) şi Total (total) pentru acest atribut. În al doilea rând am creat conceptele (nu doar prin gruparea obiectelor care au acelaşi set de atriburuti, dar şi prin compararea valorilor lor), şi apoi le-am transformat într-o latice conceptuală (fără binarizare), ceea ce a fost baza analizelor ulterioare. Cu alte cuvinte: exploatăm FCA-ul pentru a obţine concepte care conţin ontologie. FCA este relevant în acest studiu prin furnizarea informaţiilor conceptuale, ceea ce este rezultatul analizei SD cu ajutorul atributurilor prezentate anterior.

După cunoştinţa noastră, niciun studiu anterior nu a folosit implementarea FCA, cu obiectele de intrare fiind SD-uri şi atributurile de intrare fiind proprietăţile acestor dependenţe.

Pe lângă managementul complet al dependenţelor strategia noastră previne apariţia

incoerenţelor în locul curăţării celor existente. Putem concluziona că aplicaţia noastră este accesibilă, uşor de utilizat şi eficace în prevenirea incoerenţelor. Algoritmul nostru reprezintă o abordare nouă în managementului CFD şi în prelucrarea FCA, prin analiza modelelor de baze de date abstracte.

6

Extragerea datelor prin utilizarea arborilor semantici

Un rezultat major al prezentei cercetări este un instrument rapid şi precis care ajută extragerea datelor semantici din diferite tipuri de surse de date. Pentru a simplifica sarcina dificilă de a integra noi surse de date şi de a lucra cu schimbări structurale în sursele de date existente, am implementat o nouă abordare numită *RK*. *RK* este un instrument care funcţionează ca un ghid pentru scopuri de extracţie; arborele semantic generat de regulile semantice (utilizate ca date de intrare) descrie un model care reprezintă structura sursei de date.

Scopul cercetării noastre este de a simplifica procesul complicat de creare a unui extractor de date. Ne propunem să motivăm proprietarii de surse de date să ajute echipele de extragere a datelor în efectuarea sarcinilor de extragere a datelor din sursele de date. Pentru a atinge acest obiectiv am creat *RK*, un instrument care utilizează un limbaj interpretat semantic pentru a defini regulile semantice care reprezintă conceptele, inclusiv traiectoria lor structurală, relaţiile de tip părinte-copil, pluralitatea, setul de sinonime/cuvinte cheie, prioritate şi alte atribute care descriu conceptul.

Utilizăm termenul de concept aici pentru a desemna toate regulile cu acelaşi cuvânt cheie. De exemplu în cazul în care informaţiile care trebuie extrase se referă la sectorul imobiliar, posibilele concepte pot include mărimea sau preţul. Din aceste reguli *RK* generează obiecte care servesc ca noduri în construirea arborelui semantic. Arborele descrie modelul structural-semantic utilizat ca un ghid în extragerea funcţionalităţilor. Aceste metode, împreună cu funcţionalităţile de navigare sunt definite în scopuri generale, totuşi pot fi personalizate,

iar în cazul funcţiilor de navigare pot fi suprascrise dacă acest lucru este necesar. Sistemul nostru suportă extragerea datelor din surse de date structurate ierarhic, cum ar fi pagini web printre altele.

Principalele beneficii ale propunerii noastre includ potenţialul de a ajuta căutările conceptuale şi posibilitatea unor schimbări structurale fără necesitatea de a schimba aplicaţiile de extragere a datelor care extrag din surse de date ale căror structură a fost modificată.

Instrumentul nostru susţine portabilitatea, personalizarea şi extragerea conceptuală, semantică. Implementarea actuală poate fi integrată în paginile web şi are scopul de a da instrucţiuni crawler-ilor. După verificarea pentru a vedea dacă *RK* este integrat în sursa de date, extractorul poate extrage întregul set de date prin aplicarea unei singure funcţiuni. Pentru a sprijini definirea regulilor semantice am dezvoltat limbajul semantic *RK*.

Fiecare regulă reprezintă un concept şi este conştientă de conceptul părinte sau predecesor, poziţia sa în raport cu poziţia structurală a conceptului său părinte sau succesor şi poziţia structurală în raport cu valoarea conceptului. Polimorfismul este suportat, fiindcă acelaşi tip de concepte pot avea mai multe forme posibile în structură. Acest lucru înseamnă că mai multe reguli pot descrie acelaşi concept pentru a gestiona diferitele cazuri ale apariţiei sale. În astfel de cazuri acestea sunt identificate cu ajutorul indicelor. Un concept poate avea mai mulţi părinţi sau predecesori potenţiali. Sinonimele sunt susţinute din cauza divergenţelor culturale. Nu există nicio ontologie standard; denumirea conceptelor diferă de la cultură la cultură şi multe concepte au diferite denumiri în aceeaşi cultură. Instrumentul respectă diversitatea culturală, oferind posibilitatea de a defini sinonime, adică diferite cuvinte cheie pentru acelaşi concept.

Regulile sunt salvate în fişiere .rk. Personalizarea extragerii este posibilă prin combinarea acestora. Numim aceste combinaţii un tip de căutare. În mod ideal aceste reguli sunt create şi întreţinute de către proprietarii surselor de date, dar pot fi definite de către oricine care are acces la fişierele .rk sau pot fi generate în mod automat de către un modul. Extragerea funcţionează în baza regulilor structural-semantice. Dacă reclamele nu sunt considerate a fi un conţinut relevant, nicio regulă semantică nu va fi definită pentru a le suporta, astfel extractorii nici nu vor lua în considerare aceste părţi.

Regulile sunt convertite în obiecte. Relaţia de tip părinte-copil, predecesor-succesor a regulilor presupune o ierarhie. Prin crearea unui nod rădăcină abstract, care are toate conceptele cu predecesori nespecificaţi ca şi copii, un arbore este definit, unde toate nodurile corespund regulilor şi nodurile sunt relaţia de predecesor-succesor. Arborele semantic descrie planul de lucru structural-semantic pentru extractor. Modificările structurale au fost întotdeauna greu de gestionat cu ajutorul extractorilor. În cazul în care se foloseşte instrumentul nostru, în cazul modificărilor structurale doar regulile semantice trebuie păstrate. Extractorii care folosesc sursa de date nu vor simţi efectele acestei schimbări (ceea ce altfel ar face). În schimb ei pot extrage data ca şi cum nu ar exista modificări structurale.

După cum am văzut, arborele semantic descrie structura datelor. Datorită structurii ierarhice a arborelui semantic, extractorul poate restrânge în mod repetat spaţiul de căutare,

fiindcă din punct de vedere structural un concept include sub-conceptele sale, astfel când se caută sub-conceptele unui concept, reprezentarea structurală a conceptului poate servi ca o sursă a căutării.

Sarcina de extragere este îndeplinită prin simplificarea repetată a spaţiului de căutare. În cazul în care un concept nu este plural, în prima fază este găsită în interiorului părintelui sau antecedentului, iar alte cazuri nu vor fi căutate. Dacă este unul plural, se efectuează o căutare completă pentru a identifica toate apariţiile. Cu ajutorul instrumentului nostru extragerea nu este mai complicată decât un apel de funcţie. Instrumentul poate fi integrat într-o sursă de date prin copierea codului sursă şi prin crearea şi publicarea fişierelor .rk. În unele cazuri funcţiile de navigare trebuie suprascrise.

6.1 Rezultatele experimentului

Am efectuat experimente pe diferite site-uri web reale. După verificarea pentru a vedea dacă *RK* este integrat în sursa de date, extractorul a extras întregul set de date prin aplicarea unei singure funcţiuni. O funcţie suplimentară a fost utilizată pentru a trece la pagina următoare. Tabelul 6.1 include sumarul rezultatelor experimentului. Toate experimentele au fost efectuate pe un calculator echipat cu un Pentium Dual-Core CPU T4300, 2,1 GHz, cu 4 GB RAM.

	No. of rules	No. of Extracted Objects	Total Download Time (millisec)	No. of Objects/ Page	Total Extraction Time (millisec)	Extraction Time/ Page (millisec)	Extraction Time/ Object (millisec)
Test RealEstate grid page	17	11376	513748	10	38372	33.72	3.37
Test RealEstate details page	21	11376	7566787	10	265288	233.12	23.32
www.boatsandoutboards.co.uk	6	7721	2586972	20	55070	148.84	7.13
http://prep-adm.cloudapp.net/	13	160012	6840503	20	521756	65.13	3.25

Table 6.1: Rezultatele experimentului

Unul dintre site-urile web testate a fost *www.boatsandoutboards.co.uk*, din care am extras *Image* (imagine), *Link* (link), *Title* (titlu), *Properties* (proprietăţi), şi *Description* (descriere) pentru bărci. În acest caz sursa de date a conţinut 7721 bărci sau accesorii, iar timpul mediu de extragere/obiect a fost de 7,13 milisecunde. În timpul procesului de extragere instrumentul nu efectuează operaţiuni care nu sunt necesare, astfel viteza de extragere este optimă. În loc de a evalua sute sau chiar mii de expresii obişnuite (ca şi în [92]), în întreaga pagină web se evaluează doar câteva zeci de reguli structural-semantice, se alcătuieşte un arbore semantic cu ajutorul căruia toate (şi exclusiv) datele necesare sunt extrase.

Un alt site pe care am testat instrumentul nostru de extragere a fost *http : //prep − adm.cloudapp.net*, care conţine rezultatele examenelor finale ale tuturor absolvenţi ai ciclului primar din România. Din această pagină am extras cu succes *County* (judeţ), *School* (şcoala), *Name* (numele) şi notele a 160012 elevi. Pentru acest experiment am definit 13

reguli structural-semantice. Timpul mediu de extragere pe pagină a fost de 65,13 milisecunde, ceea ce corespunde la 3,25 milisecunde/elev. Acest site oferă posibilitatea de a căuta pe ea, dar dacă cineva doreşte să analizeze întregul sistem de învăţământ din România, are nevoie de toate datele de pe acest site, ceea ce nu este accesibil. Metoda noastră oferă o soluţie prin extragerea semantică a tuturor datelor necesare, ceea ce permite aplicarea unei analize suplimentare a datelor.

Nu se raportează precizie şi rechemare calculată faţă de calitatea datelor extrase deoarece nu s-au înregistrat pierderi în experimentele noastre. Precizia extragerii a fost de 100% atâta timp cât regulile structural-semantice au fost definite în mod corect. Timpul de extragere a fost egal cu timpul cumulat necesar pentru crearea arborelui semantic plus extragerea tuturor datelor prescrise cu ajutorul jQuery, mai puţin de 10 milisecunde, în medie (pe un calculator simplu cu parametrii descrise anterior). Desigur am putea utiliza un super-server şi multithreading pentru a obţine rezultate şi mai bune.

Putem concluziona că instrumentul pe care l-am creat simplifică extragerea datelor, face posibilă crearea mai multor strategii de extragere şi facilitează căutări semantice în sistemele care utilizează instrumentul. Extragerea este optimă deoarece cu ajutorul unor reguli semantice corecte, algoritmul restrânge în mod repetat spaţiul de căutare pentru a creşte performanţa căutării. În locul analizei textului am oferit o alternativă căutării ierarhice care funcţionează în mod constant în subarborele unui anumit concept pentru a găsi valorile subconceptelor în baza arborelui semantic alcătuit din seturile de reguli. Cu aplicarea regulilor corecte, datele extrase trebuie să fie corecte şi dacă acest lucru este realizat într-un mediu de colaborare, precizia poate fi menţinută pe termen lung.

Rezultatele experimentului în care s-a folosit instrumentul RK au dovedit fezabilitatea abordării noastre prin faptul că extragerea a fost amplificată din punctul de vedere al preciziei şi al vitezei.

7

Concluzii şi planuri de viitor

În această teză am prezentat modul în care FCA-ul poate fi utilizat pentru a caracteriza dependenţe funcţionale şi pentru a extrage cunoştinţe din baze de date. În viitor ne propunem să caracterizăm dependenţele funcţionale utilizând formalismul structurilor model [10], o extensie a FCA-ului clasic pentru a gestiona date complexe. Ne propunem de asemenea să dezvoltăm un software care va crea un tricontext al unui arbore XML şi va oferi dependinţele funcţionale din arborele XML. Pe de altă parte am dori să extragem, în ceea ce priveşte FCA, şi alte tipuri diferite de dependenţe care pot include un anumit set de date (nu numai în cele relaţionale şi XML, dar şi în baze de date NoSQL).

În capitolul 5 am introdus DependencyManager, abordarea noastră referitoare la managementul dependenţelor şi prevenirea contradicţiilor. În viitor ne propunem să facem această strategie şi mai utilă, ne propunem să găsim şi să analizăm candidaţi SD pe tabele ordonate transversal. Dorim de asemenea să generalizăm un set de condiţii prin utilizarea mai multor coloane în funcţiile booleene în locul unei singure şi cu utilizarea mai multor tipuri de operatori (implementarea noastră curentă consideră egalitatea ca singurul operator condiţional). Automatizarea procesului de validare ar fi utilă; această funcţie va fi probabil bazată pe reguli determinate de utilizator care ajută modulul care învaţă să determine în mod automat dacă un candidat SD este un AD sau un SD. Sistemul salvează erorile care provin din inserţii şi actualizări (update) eşuate care au fost împiedicate de limitările create pentru SD-uri; acesta va genera cunoştinţe care pot fi utilizate să determine cauza eşecului inserţiilor şi actualizărilor. În cazul în care eşecul este un model incorect acceptat, modelul trebuie respins; în caz contrar acţiunea de utilizator eronată sau funcţionalitatea incorectă poate fi detectată. SD-urile cunoscute pot fi utilizate în creşterea performanţei aplicaţiei.

Ca urmare spaţiul de stocare/performanţa de siguranţă (backup) poate fi optimizat prin crearea modelelor/şabloanelor în loc de a stoca toate cazurile de date redundante care corespund modelelor valide. Modelele valide pot fi arhivate (cache) care permit sistemele-client să deducă valori şi astfel să reducă numărul cererilor de la baza de date. Prin urmare, dimensiunea răspunsurilor bazei de date pot fi reduse, optimizând comunicarea între sistemul client, serverul aplicaţiei şi serverul bazei de date.

În capitolul 6 am introdus instrumentul nostru *RK* care ajută extragerea datelor din diferite tipuri de surse de date. În prezent setul de reguli în *RK* poate fi definit uşor, dar regulile trebuie introduse manual, deoarece niciun modul de definire automată a regulilor nu a fost încă creat. În viitor ne propunem să creăm accesorii (add-on) de browser şi aplicaţii desktop care suportă definiţii de reguli grafice generatoare de reguli RK pe baza acţiunii utilizatorului. Semantic Tree Builder (Creatorul de arbori semantici) citeşte regulile din setul de reguli şi creează un arbore semantic, creând un nod de rădăcină abstract şi alte noduri, care formează un arbore. Nodul de rădăcină există pentru a garanta faptul că structura datelor va fi un arbore, iar celelalte reprezintă regulile. Relaţiile dintre noduri formează vârfurile arborelui. În prezent nu este nicio garanţie referitor la faptul că graficul nu va conţine cicluri, ceea ce înseamnă că în cazul în care un dezvoltator creează un ciclu în mod accidental, sistemul nu îl împiedică.

În viitor ne-am propus să creăm un validator pentru arborele semantic, ceea ce ar considera arborele să fie valid doar dacă acesta este într-adevăr un arbore, şi nu conţine cicluri. Modulul Extractor al *RK* extrage reprezentarea textuală a conceptelor situate în arborele semantic. Dorim să implementăm un layer superior care va folosi Extractorul şi va suporta extragerea altor tipuri de meta-date, cum ar fi fişiere sunet, video, imagini sau alte tipuri pe baza datelor textuale extrase cu ajutorul Extractorului. Extractorul nu suportă analiza datelor textuale; layerul superior va suporta şi analiza textuală. Ideile din spatele modulelor Semantic Tree Builder, Extractor şi Navigator sunt agnostice din punct de vedere tehnologic; suportă extragerea din orice sursă de date având structura ierarhică a datelor (implementarea noastră curentă suportă extragerea de pe pagini web). Ne propunem, de asemenea, suportul altor tipuri de surse de date.

Bibliografie

[1] S. Abiteboul, R. Hull, V. Vianu. Foundations of Databases, Addison-Wesley, 1995

[2] R. Agrawal, R. Srikant. Fast algorithms for mining association rules in large databases, In VLDB '94: Proceedings of the 20th International Conference on Very Large Data Bases. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1994, 487–499

[3] S. Andrews. In-Close, a Fast Algorithm for Computing Formal Concepts, In Rudolph, Dau, Kuznetsov (Eds.), Supplementary Proc. of ICCS'09, CEUR WS 483, 2009

[4] M. Arenas, L. Libkin. A normal form for XML documents, TODS 29(1), 2004, 195–232

[5] M. Arenas, L. Libkin. An information-theoretic approach to normal forms for relational and XML data, JACM 52(2), 2005, 246–283

[6] M. Arenas, L. Libkin, W. Fan. On the Complexity of Verifying Consistency of XML Specifications, SIAM J. Comput. 38(3), 2008, 841–880

[7] J. Baixeries. A Formal Concept Analysis framework to model functional dependencies. Mathematical Methods for Learning, 2004

[8] J. Baixeries. A formal context for symmetric dependencies, In R. Medina, S. A. Obiedkov, editors, ICFCA, volume 4933 of LNCS, Springer, 2008, 90–105

[9] J. Baixeries. Lattice Characterization of Armstrong and Symmetric Dependencies (PhD Thesis), Universitat Politecnica de Catalunya, 2007

[10] J. Baixeries, M. Kaytoue, A. Napoli. Characterizing Functional Dependencies in Formal Concept Analysis with Pattern Structures, Annals of Mathematics and Artificial Intelligence, 2014

[11] P. Becker, J. Hereth, G. Stumme. ToscanaJ: An open source tool for qualitative data analysis, In V. Duquenne, B. Ganter, M. Liquiere, E. M. Nguifo, and G. Stumme, editors, Advances in Formal Concept Analysis for Knowledge Discovery in Databases, 2002.

[12] R. Belohlavek, M.Trnecka. Basic Level in Formal Concept Analysis: Interesting Concepts and Psychological Ramifications, IJCAI'13 Proc. of the Twenty-Third international joint conference on Artificial Intelligence, 2013, 1233–1239

[13] G. Beskales, I. F. Ilyas, L. Golab. Sampling the repairs of functional dependency violations under hard constraints, PVLDB, 3(1): 2010, 197–207

[14] P. Bohannon, W. Fan, F. Geerts, X. Jia, A. Kementsietsidis. Conditional functional dependencies for data cleaning, In ICDE, IEEE, 2007, 746–755

[15] P. V. Borza, O. Sabou, C. Sacarea. OpenFCA, an open source formal concept analysis toolbox, IEEE International Conference on Automation Quality and Testing Robotics AQTR, Vol 3, 2010, 1–5

[16] P. Buneman, S. Davidson, W. Fan, C. Hara, W.-C. Tan. Keys for XML, Proc. WWW, Hong Kong, China, 2001, 201–210

[17] A. Buzmakov, S. O. Kuznetsov, A. Napoli. Scalable Estimates of Concept Stability, ICFCA, 2014, 157–172

[18] S. Chakravarthy, H. Zhang. Visualization of association rules over relational DBMS, SAC '03: Proceedings of the 2003 ACM Symposium on Applied Computing, New York, NY, USA, 2003, 922–926

[19] C. H. Chang, M. Kayed, M. R. Girgis, K. F. Shaalan, A survey of web information extraction systems, IEEE Transactions on Knowledge and Data Engineering, 18(10), 2006, 1411–1427

[20] F. Chiang, R. J. Miller. Discovering data quality rules, PVLDB, 1(1): 2008, 1166–1177

[21] E. F. Codd. The relational model for database management: version 2. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1990

[22] G. Cong, W. Fan, F. Geerts, X. Jia, S. Ma. Improving data quality: Consistency and accuracy, VLDB, ACM, 2007, 315–326

[23] G. Cormode, L. Golab, F. Korn, A. McGregor, D. Srivastava, X. Zhang. Estimating the confidence of conditional functional dependencies, SIGMOD Conference, ACM, 2009, 469–482

[24] V. Crescenzi, G. Mecca, P. Merialdo. Automatic web information extraction in the roadrunner system, In Revised Papers from the HUMACS, DASWIS, ECOMO, and DAMA on ER 2001 Workshops, 2002, 264–277

[25] V. Crescenzi, D. Qiu, P. Merialdo. A Framework for Learning Web Wrappers from the Crowd, Proceedings of the 22nd WWW Conference, 2013, 261–272

[26] J. Demetrovics, L. Libkin, I. Muchnik. Functional Dependencies in Relational Databases: a Lattice Point of View, Discrete Applied Mathematics, Volume 40, Issue 2, 1992, 155–185

[27] M. H. Dunham. Data Mining: Introductory and Advanced Topics, Prentice-Hall, 2002

[28] W. W. Eckerson. Data Quality and the Bottom Line: Achieving Business Success through a Commitment to High Quality Data, Tech. rep., The Data Warehousing Institute, 2002

[29] D. W. Embley, W.Y. Mok. Developing XML documents with guaranteed "good" properties, ER 2001, 20th International Conference on Conceptual Modeling, 2001, 426–441

[30] W. Fan, F. Geerts, X. Jia, A. Kementsietsidis. Conditional functional dependencies for capturing data inconsistencies, ACM Trans. Database Syst., 33(2), 2008

[31] W. Fan, F. Geerts, L. V. S. Lakshmanan, M. Xiong. Discovering conditional functional dependencies, In ICDE, IEEE, 2009, 1231–1234

[32] W. Fan, S. Ma, Y. Hu, J. Liu, Y. Wu. Propagating functional dependencies with conditions, PVLDB, 1(1):2008, 391–407

[33] W. Fan, J. Li, S. Ma, N. Tang, W. Yu. Towards certain fixes with editing rules and master data, PVLDB, 3(1):2010, 173–184

[34] W. Fan, F. Geerts, J. Li, M. Xiong. Discovering conditional functional dependencies, IEEE Trans. Knowl. Data Eng., 23(5):2011, 683–698

[35] U. Fayyad, G. Piatetsky-Shapiro, P. Smyth. From Data Mining to Knowledge Discovery in Databases, Advances in knowledge discovery and data mining. American Association for Artificial Intelligence, 1996, 1 – 34

[36] L. Golab, H. J. Karloff, F. Korn, D. Srivastava, B. Yu. On generating near-optimal tableaux for conditional functional dependencies, PVLDB, 1(1):2008, 376–390

[37] C. F. Goldfarb. The SGML handbook, Oxford University Press, Inc., 1990

[38] J. Hereth. Relational Scaling and Databases, Proceedings of the 10th International Conference on Conceptual Structures: Integration and Interfaces LNCS 2393, Springer Verlag, 2002, 62–76

[39] J. Hereth, G. Stumme, R. Wille, U. Wille. Conceptual Knowledge Discovery - a Human-Centered Approach, Journal of Applied Artificial Intelligence, 17:2003, 281–301

[40] T. W. Hong, K. L. Clark. Using grammatical inference to automate information extraction from the web, 5th European Conference on PKDD, 2001, 216–227

[41] Y. Huhtala, J. Kärkkäinen, P. Porkka, H. Toivonen. TANE: An efficient algorithm for discovering functional and approximate dependencies, Comput. J., 42:1999, 100–111

[42] D. Ignatov, K. T. Janosi-Rancz, S. Kuznetsov. Towards a framework for near-duplicate detection in document collections based on closed sets of attributes, Acta Universitatis Sapientiae, Informatica, 2, 2009, 215–233

[43] K. T. Janosi-Rancz, V. Varga, T. Nagy Detecting XML Functional Dependencies through Formal Concept Analysis, 14th ADBIS Conference, LNCS 6295, Springer, 2010, 595–598

[44] K. T. Janosi-Rancz, V. Varga. A method for mining functional dependencies in relational database design using FCA, Studia Universitatis Babes-Bolyai, Informatica, vol. LIII, No. 1, 2008, 17–28

[45] K.T. Janosi-Rancz, A. Lajos. Semantic Data Extraction, Accepted for The 8th International Conference INTER-ENG 2014, Interdisciplinarity in Engineering, 9 - 10 October 2014, "Petru Maior" University of Tîrgu Mureş, Romania

[46] K.T. Janosi-Rancz. Finding, Managing and Inforcing CFDs and Ars via a semi-automatic learning strategy, 10th Joint Conference on Mathematics and Computer Science, May 21-25, Cluj Napoca, Romania, Studia Universitatis Babes-Bolyai, Informatica, 2014

[47] K.T. Janosi-Rancz, V. Varga. XML Schema Refinement Through Formal Concept Analysis, Studia Universitatis Babes-Bolyai, Informatica, Volume LVII, Number 3, 2012, 49–64

[48] K. T. Janosi-Rancz, V. Varga, J. Puskas, A Soft Tool for Relational Database Design using Formal Concept Analysis, 7th Joint Conference on Mathematics and Computer Science Cluj, 3-6 July 2008, Studia Universitatis Babes-Bolyai, Informatica, vol. LIII, No. 2:2008, 67–78

[49] M. Kaytoue. Mining numerical data with formal concept analysis and pattern structures (PhD Thesis), Université Henri Poincaré - Nancy, 2011

[50] A. Kirsch, M. Mitzenmacher, A. Pietracaprina, G. Pucci, E. Upfal, F. Vandin, An efficient rigorous approach for identifying statistically significant frequent itemsets, In

Proceedings of the 28th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, Providence, Rhode Island, USA, 2009.

[51] S. Kolahi, L. V. S. Lakshmanan. On approximating optimum repairs for functional dependency violations, ICDT, vol. 361 of ACM ICPS, 2009, 53–62

[52] S. Kolahi, L. Libkin. XML design for relational storage, In Proceedings of the 16th International World Wide Web Conference, 2007, 1083–1092

[53] S.O. Kuznetsov. A fast algorithm for computing all intersections of objects in a finite semilattice, Automatic documentation and Mathematical linguistics 27(5), 1993, 11–21

[54] S.O. Kuznetsov, S. A. Obiedkov. Comparing performance of algorithms for generating concept lattices, Journal of Experimental and Theoretical Artificial Intelligence, Vol. 14, Nos 2-3, 2002, 189–216

[55] A. H. F. Laender, B. A. Ribeiro-Neto, A. S. da Silva, J. S. Teixeira. A brief survey of web data extraction tools, SIGMOD Rec. 31, 2/2002, 84–93

[56] M. Lee, T. Ling, W.L. Low. Designing functional dependencies for XML, EDBT Conference, 2002, 124–141

[57] J. Liu, M. W. Vincent, C. Liu. Local XML functional dependencies, In Proceedings of the 5th ACM International Workshop on Web Information and Data Management, 2003, 23–28

[58] S. Lopes, J. M. Petit, L. Lakhal. Functional and approximate dependency mining: database and fca points of view, J. Exp. Theor. Artif. Intell., 14(2-3):, 2002, 93–114

[59] R. Medina, L. Nourine. Conditional functional dependencies: An fca point of view, In L. Kwuida and B. Sertkaya, editors, ICFCA, volume 5986 of LNCS, Springer, 2010, 161–176

[60] R. Medina, L. Nourine. A unified hierarchy for functional dependencies, conditional functional dependencies and association rules, In ICFCA, 2009, 98–113

[61] A. Mesbah, A. van Deursen, S. Lenselink. Crawling ajax-based web applications through dynamic analysis of user interface state changes, ACM Trans. Web, 6(1), 2012, 1–30

[62] I. Muslea, S. Minton, C. A. Knoblock. Active learning with multiple views. J. Artif. Intell. Res. (JAIR), 27:2006, 203–233

[63] S. Perugini, N. Ramakrishnan. Mining web functional dependencies for flexible information access, J. Am. Soc. Inform. Sci. Technol., 58:2007, 1805–1819.

[64] J. Poelmans, D. I. Ignatov, S. O. Kuznetsov, G. Dedene: Formal concept analysis in knowledge processing: A survey on applications, Expert Systems with Applications, 40(16):2013, 6538–6560

[65] U. Priss. Formal Concept Analysis in Information Science, In Cronin, B. (ed.) Annual Review of Information Science and Technology, ASIST, vol. 40, 2008

[66] K. Probst, R. Ghani, M. Krema, A. E. Fano, Y. Liu. Semi-supervised learning of attribute-value pairs from product descriptions, IJCAI., 2007, 2838–2843

[67] G. Schreiber, H. Akkermans, A. Anjewierden, R. de Hoog, N. R. Shadbolt, W. van de Velde, B. J. Wielinga. Knowledge Engineering and Management: The CommonKADS Methodology, MIT Press, 2000

[68] I. N. Md. Shaharanee, F. Hadzic, T. S. Dillon. Interestingness measures for association rules based on statistical validity, Knowledge-Based Systems, Volume 24, Issue 3, 2011, 386–392

[69] K. Techapichetvanich, A. Datta. VisAR : A new technique for visualizing mined association rules, ADMA., 2005, 88–95

[70] H. S. Thompson, D.Beech, M. Maloney, N. Mendelsohn (Editors). XML schema part 1: Structures second edition, 2004, http://www.w3.org/tr/xmlschema-1/

[71] J. D. Ullman. Principles of Database Systems. Computer Science Press, 1982

[72] P. Valtchev, R. Missaoui, R. Godin, A framework for incremental generation of closed itemsets, Discrete Applied Mathematics 156, Elsevier, 2008, 924–949

[73] V. Varga, K. T. Janosi Rancz. A Software Tool to Transform Relational Databases in Order to Mine Functional Dependencies in it Using Formal Concept Analysis, Proceedings of the Sixth International Conference on Concept Lattices and Their Applications, Olomouc, Czech Republic, 2008, 1–9

[74] V. Varga, K. T. Janosi Rancz, C. Sacarea, K. Csioban, XML Design: an FCA Point of View, Proceedings of 2010 IEEE International Conference on Automation, Quality and Testing, Robotics, Theta 17th edition, 2010, 165–170

[75] M. W. Vincent, J. Liu. Functional dependencies for XML, In Proceedings of the 5th Asian-Pacific Web Conference, 2003, 22–34

[76] M. W. Vincent, J. Liu, C. Liu. Strong functional dependencies and their application to normal forms in XML, ACM TODS, 29(3): 2004, 445–462

[77] M. W. Vincent, J. Liu, M. Mohania, On the Equivalence between FDs in XML and FDs in Relations, Acta Informatica 44(3-4), 2007, 207–247

[78] F. Vogt, R. Wille. TOSCANA - a Graphical Tool for Analyzing and Exploring Data, Graph Drawing, 1994, 226-233

[79] R. Wille. Restructuring lattice theory: an approach based on hierarchies of concepts, I. Rival (Ed.): Ordered sets, Reidel. Dordrecht-Boston, 1982, 445–470

[80] B. Ganter and R. Wille. Formal Concept Analysis: Mathematical Foundations, Springer, 1999

[81] R. Wille. Why can concept lattices support knowledge discovery in databases? Journal of Experimental and Theoretical Artificial Intelligence, 14(2-3), 2004, 81–92

[82] G. I. Webb. Discovering Significant Patterns, Machine Learning, Springer, 2007, 1–33

[83] C. Wyss, C. Giannella, E. Robertson. FastFDs: A heuristic-driven, depth-first algorithm for mining functional dependencies from relation instances, Proc. of 3rd Int. Conf. of the Data Warehousing and Knowledge Discovery, 2001

[84] T. L. Wong, W. Lam. An unsupervised method for joint information extraction and feature mining across different Web site, Data And Knowledge Engineering, 68, 2009, 107–125

[85] T. L. Wong, W. Lam. Unsupervised Extraction of Popular Product Attributes from Web Sites, 8th Asia Information Retrieval Societies Conference, 2012, 437–446

[86] W3C. XML Schema, 2004, http://www.w3.org/TR/xmlschema-0/

[87] H. Yao, H.J. Hamilton, C.J. Butz, FD_MINE: Discovering Functional Dependencies in a Database Using Equivalences, IEEE Int. Conf. on Data Mining (ICDM02), 2002, 9–12

[88] M. Yakout, A. K. Elmagarmid, J. Neville, M. Ouzzani, and I. F. Ilyas. Guided data repair, PVLDB, 4(5):2011, 279–289

[89] S. A. Yevtushenko. System of data analysis "Concept Explorer", Proceedings of the 7th national conference on Artificial Intelligence KII-2000 Russia, 2000, 127–134

[90] C. Yu, H. V. Jagadish. XML schema refinement through redundancy detection and normalization, VLDB J. 17(2):2008, 203–223

[91] Y. Zhai, B. Liu. Extracting web data using instance-based learning, World Wide Web, 10, 2007, 113–132

[92] X. Zheng, Y. Gu, Y. Li. Data extraction from web pages based on structural-semantic entropy, In Proceedings of the 21st WWW Conference, 2012, 93–102